DOCUMENT RESUME

ED 422 924 IR 057 082

AUTHOR Lippert, Susan K.; Granger, Mary J.
TITLE Peer Learning in an Introductory Programming Course.
PUB DATE 1997-00-00
NOTE 9p.; In: Proceedings of the International Academy for Information Management Annual Conference (12th, Atlanta, GA, December 12-14, 1997); see IR 057 067.
PUB TYPE Reports - Descriptive (141) -- Speeches/Meeting Papers (150)
EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS *Cooperative Learning; *Group Activities; Higher Education; Information Science Education; *Information Systems; Interaction; Introductory Courses; Peer Relationship; *Programming; Student Attitudes; Teaching Methods; Undergraduate Study

ABSTRACT

The role of Information Systems within organizations is constantly changing. Undergraduates concentrating in Information Systems need to acquire the knowledge and skills to compete in this dynamic arena. The undergraduate curriculum must not only address technical knowledge, but also communications and collaborative skills necessary for participation in the work environment. In order to provide an opportunity for students to combine these competencies, peer learning techniques were combined with the standard methods of teaching computer programming skills in an introductory programming course. These techniques allow students, as peers, to learn from and with each other. Student learning is promoted through peer/classmate interaction within a formal team setting. Learning occurs, not only from the instructor, but from and with other students; most peer learning exercises challenge the student to assume more of the learning responsibility. Peer learning techniques were implemented throughout the semester through small in-class group exercises and a larger final group programming project. This sharing of knowledge and group interaction created a better environment for learning subject matter that is often considered very difficult. Students enjoyed working with each other and their level of anxiety decreased. Although the focus of this paper is on a specific course, many of the activities can be implemented in other technical or non-technical courses. (Contains 16 references.) (Author/AEF)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED 422 924

PEER LEARNING IN AN INTRODUCTORY PROGRAMMING COURSE

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
☐ This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

Susan K. Lippert
George Washington University

Mary J. Granger
George Washington University

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
T. Case
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

The role of Information Systems within organizations is constantly changing. Undergraduates concentrating in Information Systems need to acquire the knowledge and skills to compete in this dynamic arena. The undergraduate curriculum must not only address technical knowledge but also communications and collaborative skills necessary for participation in the work environment. In order to provide an opportunity for students to combine these competencies, peer learning exercises were incorporated into an introductory programming course.

Peer learning techniques were combined with the standard methods of teaching computer programming skills. These techniques allow students, as peers, to learn from and with each other. Student learning is promoted through peer/classmate interaction within a formal team setting. Learning occurs, not only from the instructor, but from and with other students. Most peer learning exercises challenge the student to assume more of the learning responsibility.

This paper describes the use of peer learning exercises in a required undergraduate programming course. Peer learning techniques were implemented throughout the semester through small in-class group exercises and a larger final group programming project. This sharing of knowledge and group interaction created a better environment for learning subject matter that is often considered very difficult. Students enjoyed working with each other and their level of anxiety decreased. Although the focus of this paper is on a specific course, many of the activities can be implemented in other technical or non-technical courses.

INTRODUCTION

The role of Information Systems within organizations is constantly changing. Undergraduates concentrating in Information Systems need to acquire the knowledge and skills to compete in this dynamic arena. The undergraduate curriculum must not only address technical knowledge but also communications and collaborative skills necessary for participation in the work environment. In order to provide students the opportunity to combine these skills, peer learning exercises were incorporated into an introductory programming course.

Two different student populations enroll in this semester-long course. First, all undergraduate students in the Information Systems field within the School of Business and Public Management at The George Washington University are required to take this as the first course in their field of concentration. Second, all graduate students admitted to the Masters of Science in Information Systems program who do not have programming knowledge background for more advanced Information Systems material are also required to take this course. Therefore, this course is a prerequisite to all other Information Systems courses at the undergraduate level and almost all those at the graduate level. As such, it provides

IR057082

a foundation for future Information Systems courses. However, it is felt that the course can also increase communication and collaborative skills as well.

Peer learning techniques were combined with the standard methods of teaching computer programming skills. These techniques advocate students, as peers, to learn from and with each other. Student learning is promoted through peer/classmate interaction within a formal team setting. Learning occurs, not only from the instructor, but from and with other students. Most peer learning exercises challenge the student to assume more of the learning responsibility. Additionally, as students work with their peers in the problem-solving exercises, they improve their communication and collaborative skills.

Since the course's student population is a hybrid of undergraduates and graduates, participants have an opportunity to work with individuals from different age groups and varied work experiences. In addition, due to the ethnically diverse student population, students have a multi-cultural experience. Peer learning techniques were implemented through semester long small in-class group exercises and a larger final group programming project. All phases of programming were addressed - from the initial design to debugging written code.

Students in programming classes are usually required to work independently on their projects with stiff penalties for collaboration. However, software development teams within organizations work collaboratively in the design and coding of the programming project. Additionally, since there is often more than one way to problem-solve, it is beneficial for students to share their initial ideas and incorporate alternative solutions into their system design. Similarly, students are encouraged to help each other debug code. This sharing of knowledge creates a better environment for learning a subject that is often considered very difficult. Students enjoy working with one another and their level of anxiety decreases. Although the focus of this paper is on this specific course, many of the peer learning activities are appropriate for other technical or non-technical courses.

WHY PEER LEARNING?

A 1996 Worcester Polytechnic Institute (WPI) workshop sponsored by the NSF brought together computer science and information systems (CS/IS) educators for a two year project. The purpose was the exploration of and adoption of peer learning into the introductory CS/IS curriculum. Within a peer learning setting, and implementing peer learning exercises, workshop participants developed cooperative activities suitable for inclusion in their own classes. Participants were randomly assigned different teams for each in-class task. After each project was completed, outcomes were shared. Pros and cons of the development process were discussed. The process was more important than the outcome: participants wanted to understand and experience the dynamics of peer learning. Because peer learning techniques were implemented during the workshop, not just discussed, participants acquired peer learning skills through peer learning activities - a practice what you preach approach. The WPI workshop goals were to:

- "develop cooperative activities,
- disseminate the use of peer learning techniques to other faculty and schools who are currently using more traditional classroom approaches, and
- build on the experience of faculty who are using peer learning, and evaluate the effectiveness of using peer learning techniques across different educational environments." (Wills 1996)

Peer (or cooperative) learning fosters students working together as part of their own learning experience (Wills 1996). The traditional role of the instructor is expanded to include facilitation and coordination of the student groups. Students are responsible for working together toward the task completion and therefore assume greater responsibility for their own learning.

According to Rau and Heyl (1990), students who are involved with information social relations learn more than those not engaged in similar activities. Bok (1986), in his report on higher education, suggests that active discussion in the classroom provides students diversified opportunities to reason through challenging

problems. Recent AACSB guidelines (1993) emphasize team building and collaboration as a powerful learning experience. Dutt (1994) suggests that short term self-directed projects place the responsibility for learning on the students. Lippert and Granger (1996) provide examples of peer learning exercises introduced into the undergraduate curriculum.

Groccia and Miller (1996) use of Peer Learning Assistants (PLA) to facilitate the peer learning process in large class settings included a cooperative learning format. Their course design emphasized small group problem solving exercises replacing lectures and tests. Hart and Groccia (1994) suggest in-class student teams, informal gatherings of three to four students, or formal gatherings to create a collaborative environment for problem solving. Hunter (1994) indicates that cooperative learning activities show slight benefits including increased student problem-solving skills and an ability to work in groups.

Due to the perception that the introductory programming course encompasses a difficult subject to master and there is a great deal of required work, the course was selected for revision with the peer learning model. It was felt that since this course is the prerequisite for the rest of the undergraduate and graduate Information Systems curriculum, if students could improve their problem-solving, communication and collaborative skills, they would perform at a higher level in subsequent courses. Additionally, it was an attempt to make the course more interactive, less threatening, and more enjoyable.

COURSE BACKGROUND

The course selected to implement peer learning is a 14 week introductory computer programming and data structures course offered every semester. Class size is generally between 20-28 students, however in-class peer learning exercises are appropriate for almost any size class. The course consists of two components: a two hour lecture and a two hour laboratory. The lecture focuses on standard introductory programming material found in many of the typical textbooks. Students receive weekly homework assignments designed to reinforce their theoretical understanding. Students also work on weekly hands-on programming projects in the laboratory under the guidance of a graduate teaching fellow. A midterm and final examination test for understanding of theoretical concepts.

Peer learning techniques are introduced the first day of class. As an ice-breaker, students provide information about their major, year in school, computer background, and one interesting fact that others might not know. This is just one example of an ice-breaker that might be used. Each week, the majority of the class period consists of the traditional lecture method using presentation overheads. Peer learning exercises are implemented at the end of each class and in a larger project during the last 4 weeks of the semester.

WEEKLY PEER LEARNING EXERCISES

Before the peer learning model is utilized in the classroom, the instructor describes the guidelines by which the peer learning techniques are utilized. Student teams of three members are used each week for the peer learning exercise. Peer learning techniques advocate the use of student roles within small group exercises. In teams of three, students rotate into different team roles of moderator, scribe, and presenter. The role of the moderator is to facilitate the process. If the group becomes stuck on a particular aspect of the exercise, the moderator suggests alternative approaches to answer the problem. The role of the scribe is to record the results or output of the exercise so that the presenter may use the results in presenting the group's work to the class.

Students are encouraged to actively listen to one another and show respect for each other's opinions. Each week, students are randomly assigned to three-member teams, thereby enabling them to work with different class members. Students work with people of different technical backgrounds, work experience, and interpersonal capabilities and the weekly peer learning exercises require them to respond to new group members, new group dynamics, and new learning possibilities. Student roles are changed each week. Students are encouraged to assume a new role, moderator, scribe, or presenter, each week. Varying team composition forces interaction with many of the students in the

class, not just their friends or those with whom they feel comfortable. In order to endure this diversity in the teams and that each student has the opportunity to work with all other students, the instructor may predetermine and assign the team composition at the beginning of the semester.

Students, in their randomly assigned teams-of-the-day, have 10-20 minutes to complete the week's exercise. Some weekly exercises include program traces showing the flow of data through program segments or individual modules, while others include establishing a variable declaration section for the program. In another week's exercise, students may write a small 20 line program. Others address debugging issues. As more complex course material is introduced, students receive program segments with one of the following errors: syntax errors, run-time errors, or logic errors.

To ensure inclusive student participation, the instructor walks around the room and reviews work in progress while offering encouragement and suggestions. This enables the instructor to assist groups experiencing difficulties and, in addition, observe the contribution level of the students. A more formal way of evaluating participation involves feedback, often written from each team. This strategy was not implemented in this instance: the informal evaluation is a low-key approach and less threatening to the students who concentrate on the task and not the grade. The instructor reviews each completed exercise and, if errors exist, informs the team how many errors are present and encourages the members to try to resolve the issues.

The first group correctly completing the in-class exercise places the results of their collaboration on the blackboard. The presenter within that group explains the logic, the process by which the results were determined, and areas where difficulties were experienced. Discussion is opened to the class. The instructor offers additional guidelines for the presentation aspect of the peer learning exercise. Constructive criticism of the solution, beneficial to both the group and the class is acceptable. Comments such as 'that is no good' are unacceptable, as are any comments attacking the 'authors' of the solution. Only the work product can be evaluated and suggestions for improvement are encouraged. Depending upon the task, the question answered may be:

- Is the output right?
- Was the correct error located?
- Is the code generated feasible? Can it be improved? How?
- Is the code efficient? Can the quality of the code be improved? How?

Although either the type of task or the instructor dictates the relevant question, at this point, the instructor attempts to remain out of the discussion. The instructor is not viewed as a 'peer' and any input is usually construed as 'the right' answer. Some students are intimidated by the instructor's comments, even perceiving them as threatening. The role of the instructor is one of facilitator and recorder of the comments. At the end of the class discussion, the instructor may offer overall observations or suggestions. Observations may include areas where several groups experienced difficulty in completing the exercise. Suggestions for alternative logical approaches might be offered. Observations might include suggestions for improved group participation and interaction.

These peer learning exercises enable students to acquire problem solving abilities and knowledge while developing skills in group dynamics. The changing of team roles enables students to approach problems through different participatory functions. Effective communication skills and collaboration are also nurtured through the peer learning exercises. Students must communicate their ideas and strategies for problem solving with others from diverse cultural backgrounds and different programming or work experience. In this class, the age difference and the graduate-undergraduate melange should not overwhelm the undergraduate student or create a superiority complex in the graduate student - they are peers and need to work together with that understanding. The diverse mix of individuals represents the dynamics students might face in typical programming work group within industry-based organizations. These exercises facilitate teamwork and the weekly repetition strengthens problem-solving competencies. These are necessary skills for success in an Information Systems career.

During the last 4 weeks of the semester, a longer term collaborative work is assigned. This programming project is produced independent of the class lecture and laboratory time. It is an opportunity for the students to further hone those skills learned during the in-class peer learning exercises.

THE PROGRAMMING PROJECT

Students selected teams of three. The weekly in-class peer learning exercises enabled students to become familiar with their classmates' problem-solving capabilities, communication skills, and collaboration techniques. Using this knowledge, the programming project teams are able to make more informed decisions regarding the distribution of responsibilities. A goal for students is to recognize the strengths of their fellow team members and to not only utilize these strengths in the completion of the project but to learn from them as well. Although studies have shown the outcomes from self-selected teams are less creative and of lower quality than those determined by the instructor, students still prefer selecting the teams themselves. In this instance, the outcome of the 'more informed team selection' resulted in slightly higher quality of projects. However, the instructor would randomly assign students to teams in the future or select members based on previous programming experience.

The programming project consists of two phases: pseudocode development and system design implementation. During the first two week phase, students develop pseudocode for the system. Each team's pseudocode is critiqued by the professor. Logic weaknesses are identified and alternatives are suggested. In order to facilitate student thinking about particular sections of their pseudocode, detailed comments are provided. Students are encouraged to begin coding less complex sections of code–those concentrating on output formats and headings.

During the system design implementation phase, students code their design. Although this phase lasts two weeks, is it expected that students began coding during the initial phase. Another peer learning strategy involves the exchange of designs. Teams then code another team design. However, given this course's time structuring, there is not enough time to thoroughly analyze and comprehend a second design. This strategy introduces another aspect of peer learning-collaboration between teams, not just individuals.

During the last class, teams present their project output. Students discuss their system development methodology, their coding methodologies, and why they chose a particular design for the system. This forces the students to think through the process and organize their understanding into a cohesive presentation. It also allows students to learn from each other and to view other methods for solving the problem, using innovative code and presentation techniques. This part of the project reinforces their communication and collaborative skills learned in earlier exercises.

EVALUATIONS

Students evaluate each member's performance during the project. Not only do they evaluate their team member's performance, they also evaluate their own effectiveness. Groups are allocated one hundred points per group member. With three people per team and three hundred points, students allocate the points to each group member for several different evaluation categories. Categories of evaluation include contribution to the pseudocode development, problem-solving strategies, design methodologies, code generation, final product assimilation, and proper presentation of output. The quality of the evaluations themselves becomes part of the student's grade and has some effect on team members' participation grade.

Students also discuss the influence of the peer learning experience on the development of the project. Guidelines for discussing the peer learning experiences are provided. Students are instructed to observe team dynamics and attempt to understand what worked well within the groups and what did not work well. Groups are encouraged to discuss the process openly. Anecdotes relating to team interactions are shared as learning opportunities for everyone in the class. The students know this feedback will be used in future classes to improve the peer learning experience. Teams are required to share one personal narrative - a team misjudgment - and explain how to handle the situation differently. Students need to understand that this is a learning experience and they should neither be embarrassed nor make fun of others.

The project serves three purposes:

- it enables students to integrate theoretical knowledge and hands-on programming skills in a larger project.
- it enables students to use peer learning techniques in the development of a project.
- it enables students to integrate team-working skills, problem solving knowledge, systems development methodology, and communication skills.

LESSONS LEARNED

Integrating peer learning exercises requires additional instructor time. The instructor must not only restructure lecture time, but also develop peer learning exercises that can be utilized during each class. The exercises should be challenging enough but are time-restricted for administration during a 10-20 minute time period. However, feedback from students regarding the project and the peer learning experience is positive. On a recent evaluation form a student wrote, "I enjoyed the weekly in-class exercises. I learned more working with my group members. It was fun."

The instructor observed that toward the end of the semester, students appeared more willing to ask questions during the peer learning exercises. Students who were experiencing difficulty indicated a lack of understanding by directly asking for clarification from their group members. Group members provided explanation to one another. Groups also ask questions that the individuals themselves do not ask.

There are several recommendations for others attempting to use weekly peer learning exercises in the classroom. First, require students to physically move to another location in the classroom. Students tend to stay in their current location and work with their friends. The physical act of moving students from one side of the room to another appears to rejuvenate their attention and interest in the exercise. It may simply be that the act of walking restores circulation, but there is an observable difference in the attention level when students are required to change locations and teammates.

Second, ensure gender mixing of your groups. The first several weeks students tended to remain in gender similar groups. The instructor noticed the all male groups spoke louder while the all female groups appeared less vocal and less connected to the exercise. When the all female groups were separated, the less vocal females became slightly more vocal in the mixed gender groups. One female student in particular, who in the all female group tended to remain passive, became slightly more active when she was the only female in the group.

Third, change groups every week. The first three weeks the instructor suggested that individuals change groups. However, not all students complied, preferring to work with their friends. The instructor noticed an attitude of indifference developing within those teams' members. Once the instructor required the composition of the teams to change weekly and insisted that all students physically move to another location, the level of interaction increased. Since students appeared to avoid changing team members, this process needs to be facilitated. The students can self-select teams, as they did in this class or the teams can be assigned each week. One way of assigning teams is to actually develop a matrix that controls the mix of the students each week. Another way, more random, is to assign difference numbers to students each week and form team composition based on these numbers.

The feedback from students is positive. Both the laboratory programming grades and weekly theoretical homework assignment are slightly higher than in previous semesters. There are plans to incorporate a structured evaluation process of the weekly exercises. The quality of work was slightly higher than past semesters. Students are more actively engaged in the learning process then the lecture process. The in-class exercises created a less threatening team environment as the students moved to the larger team programming project.

FUTURE PLANS

The use of peer leaning techniques will continue to be implemented in this course as well as other undergraduate information systems courses. In-class peer learning exercises are being developed that incorporate the design of data flow diagrams, structure charts, entity relationship diagrams,

normalization issues and converting entity relationship diagrams to normalized relations. Used at the end of a class lecture, they effectively reinforce the lecture topic and highlight immediately any concepts students do not comprehend.

SUMMARY

As mentioned, developing these exercises creates additional work for the instructor. However, it is felt that as the students accept more responsibility for their own learning, they acquire an appreciation for not only the required course material, but for their own capabilities. Some students perceive computer programming as dry, boring and very tedious. However, the peer learning exercises engage the students in the learning process and make the material more appealing. Class participation increased and students, in general, tended to ask more questions than prior semesters. They appeared more interested in the entire experience. While there are many factors that might have caused this observable difference, it is felt that the additional instructor time and involvement was rewarded with more receptive and enthusiastic students.

REFERENCES

Box, D. (1996) *Higher Learning*, Harvard University Press, Cambridge, MA.

Chow, J., Dick, G. and Edmundson, B. (1994) "Industry Satisfaction with IS Graduates in the 1990s: An Empirical Study" *Proceedings of the Ninth Annual Conference of the International Academy for Information Management*, December 2-4, 1994, Las Vegas, Nevada, 153-174.

Cooper, J.L. (1995) "Cooperative Learning And Critical Thinking," *Teaching Of Psychology*, February 1995, Vol. 22, No. 1, 7-9.

Cougar, J.D., Davis, G.B, Dologite, D.G., Feinstein, D.L., Gorgone, J.T., Jenkins, A. M., Kasper, G. M., Little, J. Cu., Longenecker, H.E., and Valacich, J.S. (1995) "IS'95 Guideline for Undergraduate IS Curriculum," *MIS Quarterly*, September 1995, Vol. 19, No. 3, 341-359.

Dutt, J. S. (1994) "A Cooperative Learning Approach to Teaching an Introductory Programming Course" *Proceedings of the Ninth Annual Conference of the International Academy for Information Management*, December 2-4, 1994, Las Vegas, Nevada, 225-232.

Fisher, B. A. (1980) *Small Group Decision Making* (2nd ed.). New York: McGraw-Hill.

Fowell, S.P. and Levy, P. (1995) "Computer-Mediated Communication In The Information Curriculum: An Initiative In Computer-Supported Collaborative Learning," *Education for Information*, 13, 193-210.

Groccia, J.E. and Miller, J.E. (1996) "Collegiality In The Classroom: The Use of Peer Learning Assistants In Collaborative Learning In Introductory Biology", *Innovative Higher Education*, Vol 21, No. 2, 87-100.

Hart, F. L. and Groccia, J. E. (1994) "An Integrated, Cooperative Learning Oriented Freshman Civil Engineering Course: Computer Analysis in Civil Engineering" *Proceedings of the Nineteenth International Conference on Improving College Teaching*, February 18-22, 1994.

Hunter, C.L. (1996) "Student as Teacher: Cooperative Learning Strategies in the Community College Classroom," *Technical Report on Issues of Education at Community Colleges: Essays By Fellows* In the Mid-Career Fellowship Program At Princeton University, Brookdale Community College, June 1996.

Johnson, D.W., Johnson, R.T., and Smith, K.A. (1991) *Active Learning: Cooperation In The College Classroom*. Edina, Minnesota: Interaction Book Company.

Keeler, C. M. and Anson, R. (1995) "An Assessment of Cooperative Learning Used For Basic Computer Skills Instruction In The College Classroom," *J. Educational Computing Research*, Vol. 12, No. 4, 379-393.

Lippert, S. K. and Granger, M.J. (1996) "Redesigning An Undergraduate Information Systems Course With Individual And Team Exercises" *Proceedings of the Eleventh Annual International Academy of Information Management*, December13-15, 1996, 26-35.

Rau , W. and Heyl, B.S. (1990) "Humanizing The College Classroom: Collaborative Learning And Social Organization Among Students," *Teaching Sociology*, Apr 1990, Vol. 18, 141-155.

Strommen, E. (1995) "Cooperative Learning," *Electronic Learning*, Mar 1995, Vol. 14, No. 6, 24-35.

Wills, C. (1996) Workshop On Application of Peer Learning To The Introductory Computer Science Curriculum, supported by the NSF, Award #DUE95-54706, Worcester, MA, (June 1996).